Universidad Politécnica de Madrid–Escuela Técnica Superior de Ingenieros Industriales
Grado en Ingeniería en Tecnologías Industriales. Curso 2015-2016-3º
Matemáticas de Especialidad–Ingeniería Eléctrica

Programación Entera

José Luis de la Fuente O'Connor
jldelafuente@etsii.upm.es
joseluis.delafuente@upm.es

Índice

Introducción

- La programación entera se ocupa de los problemas de optimizar una función de diversas variables sujeta a condiciones de igualdad y/o desigualdad, restringiéndose todas o alguna de esas variables a tomar valores enteros.

- Sus orígenes se remontan a los años 50 del siglo XX. El pionero fue Ralph Gomory, EE.UU. 1929.

- Las áreas de aplicación práctica son muchas donde hay que asignar recursos sólo disponibles en cantidades discretas: distribución de mercancías, programación de la producción en factorías, secuenciación de maquinaria en procesos industriales y productivos, asignación de grupos generadores de energía eléctrica, cadenas de suministro, logística, programación de rutas de vuelos, etc.

- La formulación general del problema lineal de Programación Entera es

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{Z}^n\ \boldsymbol{y}\in\mathbb{R}^p}{\text{maximizar}} \quad & \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{h}^T\boldsymbol{y} \\ \text{sujeta a} \quad & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{G}\boldsymbol{y} \leq \boldsymbol{b} \\ & \boldsymbol{x},\ \boldsymbol{y} \geq \boldsymbol{0}. \end{aligned}$$

  Este problema es un programa entero mixto.

- Al conjunto $S = \{\boldsymbol{x} \in \mathbb{Z}^n,\ \boldsymbol{y} \in \mathbb{R}^p : \boldsymbol{A}\boldsymbol{x} + \boldsymbol{G}\boldsymbol{y} \leq \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{y} \geq \boldsymbol{0}\}$ se le denomina región factible.

- Un punto $[\boldsymbol{x}^T, \boldsymbol{y}^T]^T \in S$ se denomina factible. Al punto $[\boldsymbol{x}^{*T}, \boldsymbol{y}^{*T}]^T \in S$, tal que

$$\boldsymbol{c}^T\boldsymbol{x}^* + \boldsymbol{b}^T\boldsymbol{y}^* \geq \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{h}^T\boldsymbol{y}, \quad \text{para todo } [\boldsymbol{x}^T, \boldsymbol{y}^T]^T \in S,$$

  se le denomina solución óptima del problema.

- Al problema en el que todas las variable son enteras,

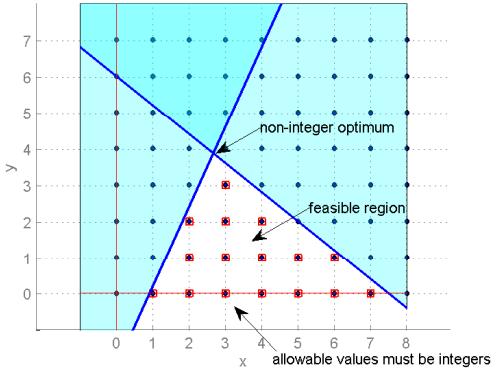
$$\begin{aligned} \max_{\boldsymbol{x} \in \mathbb{Z}^n} \ & \boldsymbol{c}^T \boldsymbol{x} \\ \text{s. a} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0}, \end{aligned}$$

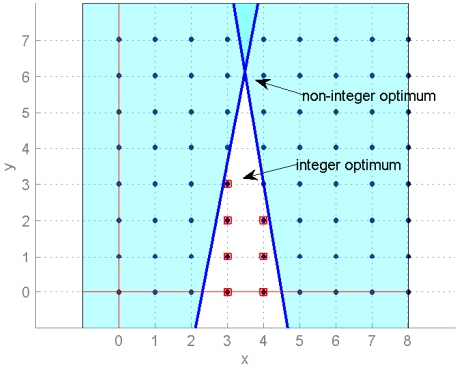
  se le conoce como programa entero puro, programa lineal en variables enteras o, simplemente, programa entero. Un caso particular de programa entero es aquel en el que las variables sólo pueden tomar valores 0 ó 1.

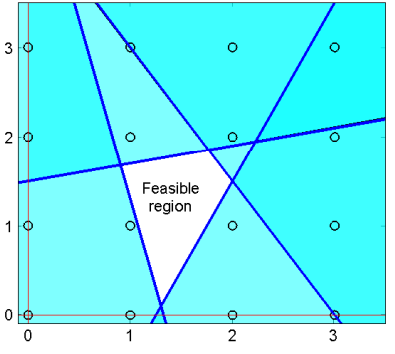
- En programación entera no existe un método universal, como el simplex de programación lineal, que obtiene la solución sirviéndose de las propiedades de convexidad del problema a resolver.

- Esto es así porque en programación entera la convexidad desaparece, no pudiéndose utilizar por tanto la noción de gradiente para caracterizar y buscar el óptimo de un problema, haciéndose necesario emplear métodos de resolución específicos del aspecto combinatorio de las variables enteras.

- Es más, los programas enteros, como muestran los dos primeros ejemplos de la figura, si se quiere minimizar $-y$, no tienen por qué tener el óptimo cerca del óptimo del programa con variables continuas, ni siquiera el óptimo es un entero próximo a ese óptimo continuo (la segunda figura).







- También puede que la región factible no registre ni un sólo entero factible, existiendo dicho espacio en el sentido tradicional.

Ejemplos

Gestión de un servicio hospitalario

- En un determinado servicio de asistencia hospitalaria hay i enfermos a la espera de una operación. Cada enfermo necesita una operación de duración D_i.

- Dada una disponibilidad de cirujanos, la suma de las duraciones de las operaciones que pueden hacerse cada día j del período de estudio es igual a T_j.

- Se trata de minimizar una función económica que refleja la suma de las penalizaciones por esperas de los diferentes enfermos (esta penalización es una función lineal creciente de la espera).

- El problema se formula de la siguiente manera:

$$\begin{aligned} \text{max.} \quad & \sum_i \sum_j c_{ij} x_{ij} \\ \text{s. a} \quad & \sum_i D_i x_{ij} \leq T_j && \text{para todo } j \\ & \sum_j x_{ij} = 1 && \text{para todo } i \\ & x_{ij} = 0 \text{ ó } 1 && \text{para todo } i \text{ y } j\,. \end{aligned}$$

La variable de decisión x_{ij} es 1 si el enfermo i se le opera el día j y 0 en el caso contrario.

El problema de la mochila

- Este problema es el más clásico de la programación entera.

- Trata de un transportista que dispone de un vehículo con capacidad volumétrica de transporte b, en el que se pueden alojar diferentes objetos j de volúmenes a_j y valores económicos c_j, y que quiere maximizar el valor de lo transportado en cada viaje.

- Su formulación es:

$$\text{max.} \sum_j c_j x_j$$

$$\text{s. a} \sum_j a_j x_j \leq b$$

$$x_j = 0 \text{ ó } 1.$$

Resolución gráfica del problema

- Recordemos de programación lineal que las condiciones de

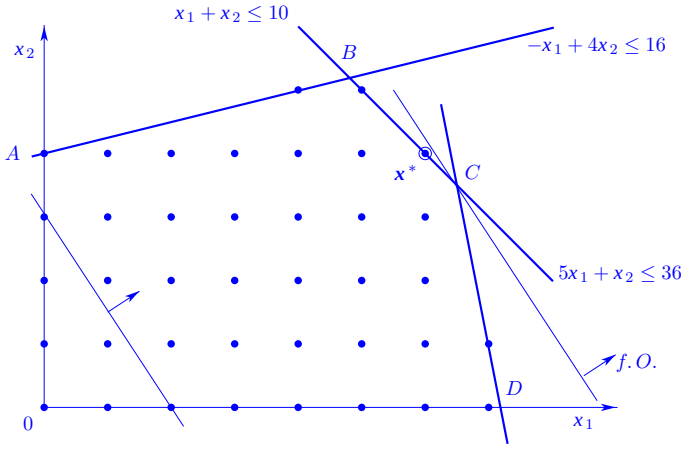
$$\begin{array}{ll} \underset{\boldsymbol{x}\in\mathbb{Z}^n\ \boldsymbol{y}\in\mathbb{R}^p}{\text{maximizar}} & \boldsymbol{c}^T\boldsymbol{x}+\boldsymbol{h}^T\boldsymbol{y} \\ \text{sujeta a} & \boldsymbol{A}\boldsymbol{x}+\boldsymbol{G}\boldsymbol{y}\leq\boldsymbol{b} \\ & \boldsymbol{x},\ \boldsymbol{y}\geq\boldsymbol{0} \end{array}$$

  en el caso en que $\boldsymbol{x}\in\mathbb{R}^n$ y $\boldsymbol{y}\in\mathbb{R}^p$, definen un politopo (poliedro si es acotado) en uno de cuyos puntos extremos la función objetivo $\boldsymbol{c}^T\boldsymbol{x}+\boldsymbol{h}^T\boldsymbol{y}$ alcanza el máximo.
  En general, desafortunadamente, las variables x_j es ese punto extremo no tienen por qué ser enteras.

- El politopo contiene todas las soluciones en las que $\boldsymbol{x}$ es entero. Gráficamente se puede determinar la solución mediante el punto extremo óptimo considerando las variables como continuas y luego, desplazando $\boldsymbol{c}^T\boldsymbol{x}+\boldsymbol{h}^T\boldsymbol{y}$ hacia dentro del politopo, encontrando el primer punto $[\boldsymbol{x}^T,\ \boldsymbol{y}^T]^T$ en el que $\boldsymbol{x}$ sea entero.

Ejemplo Consideremos el problema de programación entera

$$\begin{aligned} \text{max. } & \tfrac{3}{2}x_1 + x_2 \\ \text{s. a } & -x_1 + 4x_2 \leq 16 \\ & x_1 + x_2 \leq 10 \\ & 5x_1 + x_2 \leq 36 \\ & x_1, x_2 \geq 0 \\ & x_1 \text{ y } x_2 \text{ enteras.} \end{aligned}$$



- Sin tener en cuenta la condición de que las variables han de ser enteras, la región factible es el poliedro convexo que generan los vértices 0, A, B, C y D de la figura.

- El valor máximo de la función objetivo en este poliedro se alcanza en C. Como la auténtica región factible la constituyen los puntos interiores enteros de ese poliedro, el máximo del problema se alcanza en el punto $x^* = [6, 4]^T$. •

Índice

Relajaciones en la formulación

- La idea es tratar de sustituir el programa entero original por otro más fácil de resolver en el que se relajan algunas de las condiciones.
  La solución de éste proporciona una cota de la función objetivo del problema original.

- Mediante un proceso iterativo que integre estas relajaciones y la resolución de los subproblemas se puede acotar cada vez más la función objetivo hasta obtener la solución final.

- Una relajación del programa entero

(PE) $\qquad\qquad$ max. $\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in S\}$

la constituye cualquier subproblema de la forma

(PR) $\qquad\qquad$ max. $\{z_{PR}(\boldsymbol{x}) : \boldsymbol{x} \in S_{PR}\}$

con las propiedades siguientes:

$$S_{PR} \supseteq S \qquad \text{y} \quad \boldsymbol{c}^T\boldsymbol{x} \leq z_{PR}(\boldsymbol{x}) \quad \text{para } \boldsymbol{x} \in S.$$

Proposición Si el problema (PR) no es factible, tampoco lo es (PE). Si (PR) es factible, los valores de las funciones objetivo cumplen que $z_{PE} \leq z_{PR}$.

Relajación lineal

- La relajación más inmediata resulta de omitir la condición de que las variables han de ser enteras.

- Si el conjunto o región factible del programa entero es $S = P \cap \mathbb{Z}^n$, la relajación lineal es

$$(PL) \qquad \text{max. } \left\{ \boldsymbol{c}^T \boldsymbol{x} : \boldsymbol{x} \in P \right\}, P = \{ \boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0} \}$$

Proposición Si (PL) es no acotado, el programa entero (PE) es no factible o no acotado.

Algoritmo tipo

Paso 0 – Inicialización Hacer $i = 1$, $w^* = \infty$ y $z^* = -\infty$. Escoger un $S_R^{(1)} \supseteq S$ tal que $z_R^{(1)} \geq \boldsymbol{c}^T \boldsymbol{x}$ para $\boldsymbol{x} \in S$.

Paso 1 – Relajación Resolver el problema PE relajado:

$$\left(R^{(i)}\right) \qquad \text{max.} \left\{ z_R^{(i)}(\boldsymbol{x}) : \boldsymbol{x} \in S_R^{(i)} \right\}.$$

Paso 2 – Comprobación de óptimo Sea $\boldsymbol{x}^{(i)}$ la solución del problema anterior. Si $\boldsymbol{x}^{(i)} \in S$: PARAR; ésta es la solución óptima de (PE) con función objetivo $w^* = \boldsymbol{c}^T \boldsymbol{x}^{(i)} = z^*$; si no, seguir.

Paso 3 – Mejora de la solución Hacer $w^* = z_R^{(i)}$, $z^* = \boldsymbol{c}^T \boldsymbol{x}^{(i)}$ si $\boldsymbol{x}^{(i)} \in S$ y $i \leftarrow i + 1$. Escoger $S_R^{(i+1)}$ tal que $S \subseteq S_R^{(i+1)} \subseteq S_R^{(i)}$ y $z_R^{(i+1)}(\boldsymbol{x})$ tal que $\boldsymbol{c}^T \boldsymbol{x} \leq z_R^{(i+1)}(\boldsymbol{x}) \leq z_R^{(i)}(\boldsymbol{x})$ para $\boldsymbol{x} \in S$ con $S_R^{(i+1)} \neq S_R^{(i)}$ o $z_R^{(i+1)}(\boldsymbol{x}) \neq z_R^{(i)}(\boldsymbol{x})$.
Ir al paso 1

Índice

Algoritmo de planos o cortes de Gomory

- Este algoritmo, formulado por Gomory en 1960, es uno de los primeros que se emplearon para resolver el problema general de programación entera.

- Consideraremos que el problema que resuelve –notación original– es

$$\text{max. } \left\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in S^{(e)}\right\}, \quad \text{donde } S^{(e)} = \{\boldsymbol{x} \in \mathbb{Z}^n : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}\},$$

y está escrito de la siguiente forma:

$$(PE) \qquad \text{max. } \left\{x_0 : \left[x_0,\ \boldsymbol{x}^T\right]^T \in S^{(0)}\right\},$$

donde

$$S^{(0)} = \left\{x_0 \in \mathbb{Z}, \boldsymbol{x} \in \mathbb{Z}^n : x_0 - \boldsymbol{c}^T\boldsymbol{x} = 0, \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\right\}.$$

- Supondremos que se dispone de una solución y base óptimas de la relajación lineal del programa entero.

- El problema se puede escribir así:

$$\begin{aligned} &\text{maximizar } x_0 \\ &\text{s. a} \qquad x_{B_i} + \sum_{j \in H} \bar{a}_{ij} x_j = \bar{a}_{i0} \quad \text{para } i = 0, 1, \ldots, m \\ &\qquad\quad x_{B_0} \in \mathbb{Z},\ x_{B_i} \in \mathbb{Z}_+, \quad \text{para } i = 1, \ldots, m \\ &\qquad\quad x_j \in \mathbb{Z}_+ \text{ para } j \in H, \end{aligned} \tag{1}$$

$x_0 = x_{B_0}$, x_{B_i}, $i = 1, \ldots, m$, son las variables básicas y x_j, $j \in H \subset N = \{1, \ldots, n\}$, las no básicas $\left(x_B = B^{-1}b - B^{-1}Nx_N \text{ y } z = c_B B^{-1} b + (c_N - c_B B^{-1} N) x_N.\right)$.

- Como esta base es factible en el programa primal y dual se cumple que $\bar{a}_{i0} \geq 0$ para $i = 1, \ldots, m$, y que $\bar{a}_{0j} \geq 0$, para $j \in H$.

- Suponiendo existe un i tal que $\bar{a}_{i0} \notin \mathbb{Z}$, se tiene el siguiente resultado.

Proposición Corte Fraccionario de Gomory Si $\bar{a}_{i0} \notin \mathbb{Z}$, $\sum_{j \in H} f_{ij} x_j = f_{i0} + x_{n+1}$, $x_{n+1} \in \mathbb{Z}_+$, es una desigualdad válida en $S^{(0)}$, donde $f_{ij} = \bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor$ para $j \in H$ y $f_{i0} = \bar{a}_{i0} - \lfloor \bar{a}_{i0} \rfloor$.

- **Ejemplo** Considérese el programa entero

$$
\begin{array}{lll}
(PE) & \text{maximizar} & 7x_1 + 2x_2 \\
& \text{s. a} & -x_1 + 2x_2 \leq 4 \\
& & 5x_1 + x_2 \leq 20 \\
& & -2x_1 - 2x_2 \leq -7 \\
& & \boldsymbol{x} \in \mathbb{Z}^2_+.
\end{array}
$$

- Introduzcamos las variables de holgura x_3, x_4 y x_5. El programa[1] queda:

$$
\begin{array}{lll}
\text{maximizar} & 7x_1 + 2x_2 & \\
\text{s. a} & -x_1 + 2x_2 + x_3 & = 4 \\
& 5x_1 + x_2 + x_4 & = 20 \\
& -2x_1 - 2x_2 + x_5 & = -7 \\
& x_1, x_2 \in \mathbb{Z}_+; \quad x_3, x_4, x_5 \geq & 0.
\end{array}
$$

[1] Aunque las variables de holgura serán enteras, no hay ninguna necesidad de restringirlas a tomar valores enteros.

- Restringiendo las variables x_3, x_4 y x_5 a tomar valores enteros el problema es:

$$\begin{array}{lrcl}
\text{maximizar } x_0 & & & \\
\text{s. a} & x_0 - 7x_1 - 2x_2 & = & 0 \\
& -\ x_1 + 2x_2 + x_3 & = & 4 \\
& 5x_1 + x_2 + x_4 & = & 20 \\
& -2x_1 - 2x_2 + x_5 & = & -7 \\
& x_0 \in \mathbb{Z}, x_j \in \mathbb{Z}_+ \text{ para } j = 1, \ldots, 5. & &
\end{array}$$

- La solución de la relajación lineal de este programa entero es

$$\begin{array}{rcl}
x_0 + \frac{3}{11}x_3 + \frac{16}{11}x_4 & = & \frac{332}{11} \\
x_1 - \frac{1}{11}x_3 + \frac{2}{11}x_4 & = & \frac{36}{11} \\
x_2 + \frac{5}{11}x_3 + \frac{1}{11}x_4 & = & \frac{40}{11} \\
\frac{8}{11}x_3 + \frac{6}{11}x_4 + x_5 & = & \frac{75}{11}.
\end{array}$$

El corte de Gomory que se puede obtener de la primera ecuación

$$\frac{3}{11}x_3 + \frac{5}{11}x_4 = \frac{2}{11} + x_6, \qquad x_6 \in \mathbb{Z}_+.$$

Paso 0 – Inicialización Hacer

$$S_R^{(1)} = \left\{x_0 \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^n : x_0 - \boldsymbol{c}^T\boldsymbol{x} = 0, \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\right\},$$

$i = 1$ y $z_R^{(1)} = x_0$.

Paso 1 – Resolución de la relajación lineal Resolver el programa entero relajado

$$(PR^{(i)}) \qquad \text{max.} \left\{x_0 : [x_0, \boldsymbol{x}^T]^T \in S_R^{(i)}\right\}.$$

Si $PR^{(i)}$ es factible y tiene solución óptima, $[x_0^{(i)}, \left(\boldsymbol{x}^{(i)}\right)^T]^T$, continuar.

Paso 2 – Comprobación de óptimo Si $\boldsymbol{x}^{(i)} \in \mathbb{Z}_+^n$, ésta es la solución óptima del programa entero.

Paso 3 – Comprobación de no factibilidad Si $PR^{(i)}$ no es factible, el programa entero original no es factible.

Paso 4 – Adición de un corte de Gomory Escoger una fila $x_{B_i} + \sum_{j \in H^{(i)}} \bar{a}_{ij}^{(i)} x_j = \bar{a}_{i0}^{(i)}$ con $\bar{a}_{i0}^{(i)} \notin \mathbb{Z}$. Hacer

$$\sum_{j \in H^{(i)}} f_{kj} x_j - x_{n+i} = f_{k0}, \quad k = 1, \ldots, m, \; x_{n+i} \in \mathbb{Z}_+,$$

el corte de Gomory de esa fila. Hacer

$$S_R^{(i+1)} = S_R^{(i)} \cap \left\{x_0 \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^{n+i} : \sum_{j \in H^{(i)}} f_{kj} x_j - x_{n+i} = f_{k0}, \boldsymbol{x} \geq \boldsymbol{0}\right\}.$$

Hacer $i \leftarrow i + 1$ e ir al paso 1.

Algoritmo de los planos o cortes de Gomory

- Cuando se añade un corte, la nueva base, que incluye x_{n+i} como variable básica, es factible del dual.
- La factibilidad del primal se incumple sólo en que x_{n+i} es negativa.
- Para reoptimizar el problema lo más adecuado es usar el método dual del simplex.
- Si $PR^{(i)}$ es no acotado, el programa entero, PE, es no acotado o no factible.

- **Ejemplo, continuación.** La última solución de $PR^{(1)}$ a la que hacíamos referencia era

$$\begin{aligned} x_0 \quad\quad + \tfrac{3}{11}x_3 + \tfrac{16}{11}x_4 \quad\quad &= \tfrac{332}{11} \\ x_1 \quad - \tfrac{1}{11}x_3 + \tfrac{2}{11}x_4 \quad\quad &= \tfrac{36}{11} \\ x_2 + \tfrac{5}{11}x_3 + \tfrac{1}{11}x_4 \quad\quad &= \tfrac{40}{11} \\ \tfrac{8}{11}x_3 + \tfrac{6}{11}x_4 + x_5 &= \tfrac{75}{11}, \end{aligned}$$

donde $x_3 = x_4 = 0$.
Es decir $\left[x_0^{(1)}, \left(\boldsymbol{x}^{(1)}\right)^T\right] = [x_0^{(1)}, x_1^{(1)}, \ldots, x_5^{(1)}] = \left[\tfrac{332}{11}, \tfrac{36}{11}, \tfrac{40}{11}, 0, 0, \tfrac{75}{11}\right]$.

- El corte de Gomory respecto de la tercera ecuación es $\tfrac{5}{11}x_3 + \tfrac{1}{11}x_4 = \tfrac{7}{11} + x_6$, $x_6 \in \mathbb{Z}_+$. Añadiéndolo a $PR^{(1)}$ se obtiene $PR^{(2)}$ cuya solución es

$$\begin{aligned} x_0 \quad\quad + \tfrac{7}{5}x_4 \quad\quad + \tfrac{3}{5}x_6 &= \tfrac{149}{5} \\ x_1 \quad\quad + \tfrac{1}{5}x_4 \quad\quad - \tfrac{1}{5}x_6 &= \tfrac{17}{5} \\ x_2 \quad\quad\quad\quad + x_6 &= 3 \\ \tfrac{2}{5}x_4 + x_5 + \tfrac{8}{5}x_6 &= \tfrac{29}{5} \\ x_3 + \tfrac{1}{5}x_4 \quad\quad - \tfrac{11}{5}x_6 &= \tfrac{7}{5}. \end{aligned}$$

- El corte de Gomory correspondiente a la primera de estas ecuaciones es $\frac{2}{5}x_4 + \frac{3}{5}x_6 = \frac{4}{5} + x_7$, $x_7 \in \mathbb{Z}_+$. Añadiéndolo a $PR^{(2)}$ se obtiene $PR^{(3)}$, de solución:

$$\begin{array}{rrrrrrrrr}
x_0 & & + & x_4 & & & + & x_7 & = 29 \\
& x_1 & + & \frac{1}{3}x_4 & & & - & \frac{1}{3}x_7 & = \frac{11}{3} \\
& & x_2 & - \frac{2}{3}x_4 & & & + & \frac{5}{3}x_7 & = \frac{5}{3} \\
& & & \frac{5}{3}x_4 & + x_5 & & - & \frac{11}{3}x_7 & = \frac{13}{3} \\
& & x_3 & - \frac{2}{3}x_4 & & & + & \frac{8}{3}x_7 & = \frac{11}{3} \\
& & & \frac{2}{3}x_4 & & + x_6 & - & \frac{5}{3}x_7 & = \frac{4}{3}\,.
\end{array}$$

- El siguiente corte se elige de la fila $x_2 - \frac{2}{3}x_4 + \frac{5}{3}x_7 = \frac{5}{3}$. Es[2]

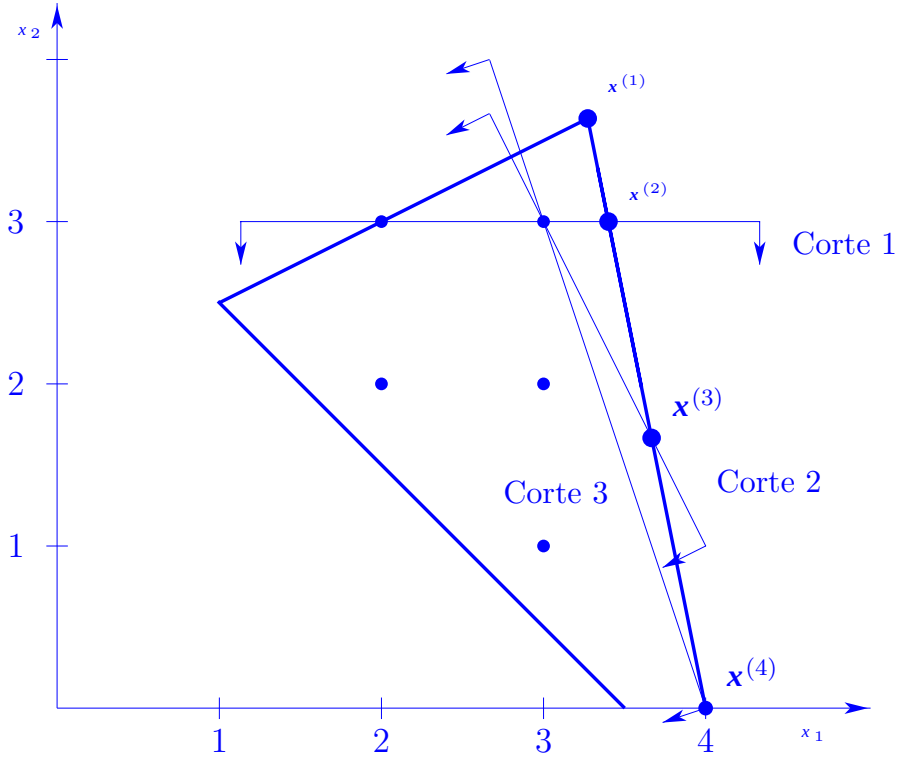
$$\frac{1}{3}x_4 + \frac{2}{3}x_7 = \frac{2}{3} + x_8, \qquad x_8 \in \mathbb{Z}_+.$$

- La solución óptima de $PR^{(4)}$ –también del problema original–, es

$$\left[x_0^{(4)}, x_1^{(4)}, \ldots, x_6^{(4)}, x_7^{(4)}, x_8^{(4)}\right] = [28, 4, 0, 8, 0, 1, 3, 1, 0].$$

[2] Recordemos $\lfloor -\frac{2}{3} \rfloor = -\frac{2}{3} - (-\frac{3}{3}) = \frac{1}{3}$.

- En términos de las variables originales, los tres cortes añadidos al problema son $x_2 \leq 3$, $2x_1 + x_2 \leq 9$ y $3x_1 + x_2 \leq 12$.



Índice

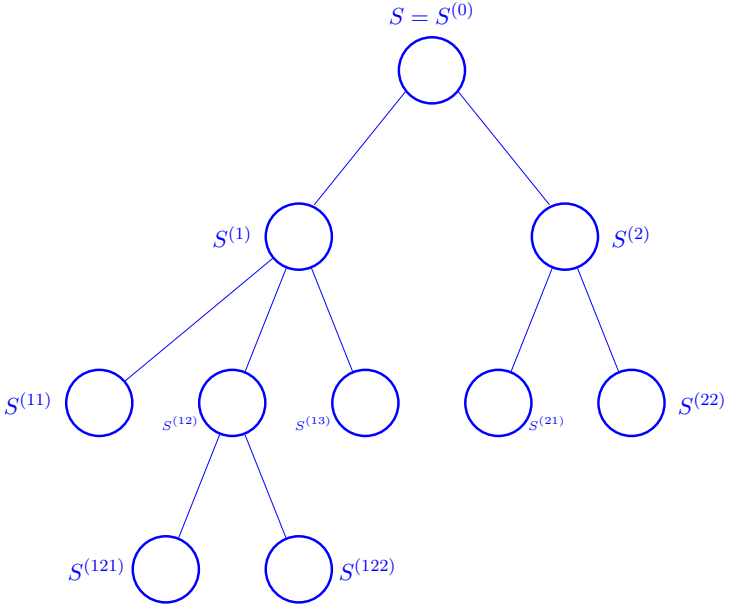
Algoritmos de ramificación y acotamiento o Branch and Bound

- La idea que los anima es dividir la región factible, S, como si las variables que la definen fuesen las ramas de un árbol e ir acotando la solución del estudio de esas ramas.

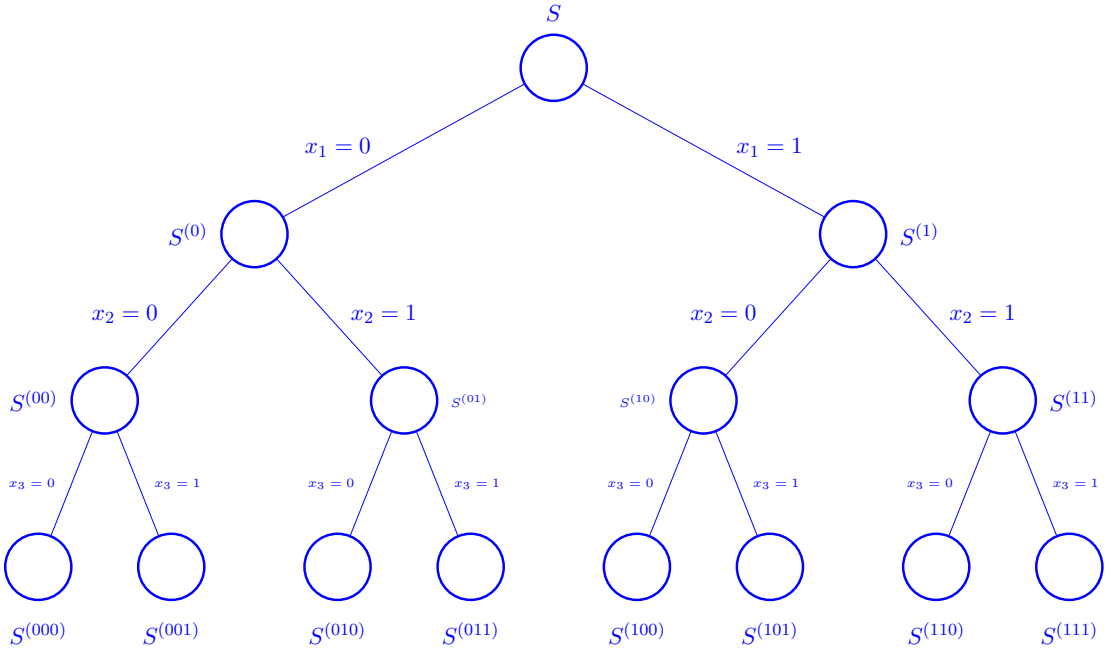
- El conjunto $\{S^{(i)} : i = 1, \ldots, k\}$ se denomina división de la región factible, S, de un programa entero, si $\cup_{i=1}^{k} S^{(i)} = S$.

- Una división se denomina partición si $S^{(i)} \cap S^{(j)} = \emptyset$ para $i,\ j = 1, \ldots, k$, $i \neq j$.

- La división se suele hacer de forma recursiva así:



- Cuando $S \subseteq \mathbb{Z}^n$ y las variables han de ser 0 ó 1, la forma más simple de hacer la división recursiva apuntada es la indicada en la figura.



- En la práctica estos métodos necesitan algún procedimiento para eliminar ramas del árbol de búsqueda que se forma –podarlo–.

Proposición El árbol de búsqueda se puede podar a partir del nudo correspondiente a $S^{(i)}$ si se cumplen cualquiera de las siguientes condiciones:

1. No factibilidad: $S^{(i)} = \emptyset$.
2. Optimalidad: se ha llegado al óptimo de $PE^{(i)}$.
3. Existencia de un valor dominante $z_{PE}^{(i)} \leq z_{PE}$.

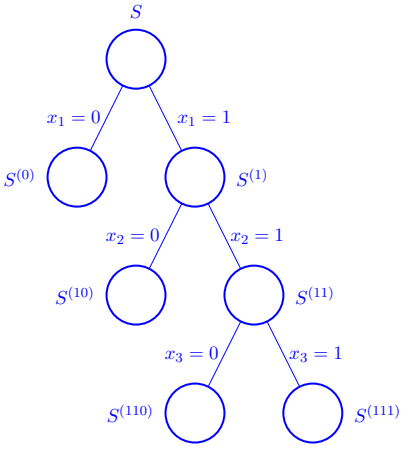
- Para no tener que resolver $PE^{(i)}$ se resuelve su relajación lineal; esto es $PL^{(i)}$ con $S^{(i)} \subseteq S_{PL}^{(i)}$ y $z_{PL}^{(i)}(\boldsymbol{x}) \geq \boldsymbol{c}^T\boldsymbol{x}$ para $\boldsymbol{x} \in S^{(i)}$.

Proposición El árbol de búsqueda puede podarse a partir del nudo correspondiente a $S^{(i)}$ si se cumple cualquiera de las condiciones siguientes:

1. El programa $PL^{(i)}$ no es factible.
2. El óptimo $\boldsymbol{x}_{PL}^{(i)}$ de $PL^{(i)}$ cumple que $\boldsymbol{x}_{PL}^{(i)} \in S^{(i)}$ y que $z_{PL}^{(i)} = \boldsymbol{c}^T\boldsymbol{x}_{PL}^{(i)}$.
3. $z_{PL}^{(i)} \leq z_{PE}$, donde z_{PE} es alguna solución factible de PE.

- **Ejemplo** Considérese el programa entero

$$
\begin{array}{lll}
(Z_{PE}) & \text{maximizar } -100x_1 + 72x_2 + 36x_3 & \\
& \text{s. a} \quad -2x_1 + x_2 \leq 0 & \\
& \qquad -4x_1 + x_3 \leq 0 & \\
& \qquad x_1 + x_2 + x_3 \geq 0 & \\
& \qquad \boldsymbol{x} \in \mathbb{Z}^3(0,1). &
\end{array}
$$



La división de este ejemplo y su árbol de búsqueda son los de la figura.

- La solución de la relajación lineal es $[1/2,\ 1,\ 1]^T$; la f.o. 58.

- Usando la última proposición para podar el árbol de búsqueda,

$$S^{(0)} = \{\boldsymbol{x} \geq \boldsymbol{0} : x_1 = 0,\ x_2 \leq 0,\ x_3 \leq 0,\ x_2 + x_3 \geq 1\} = \emptyset,$$

por lo que el nudo $S^{(0)}$ se desecha.

- La condición de óptimo de los nudos $S^{(110)}$ y $S^{(111)}$ se cumple pues estos conjuntos contienen las soluciones $[1,\ 1,\ 0]^T$ y $[1,\ 1,\ 1]^T$, respectivamente.

- Como, $z_{PE}^{(110)} < z_{PE}^{(111)} = 8$, se tendrá que $\underline{z}_{PE} = z_{PE}^{(111)} = 8$.

- Aplicando la tercera condición de la mencionada proposición al nudo $S^{(10)}$ se ve que $z_{PL}^{(10)} = -64 < \underline{z}_{PE}$, por lo que se desecha.

- Como resultado final se obtiene que $\boldsymbol{x} = [1,\ 1,\ 1]^T$ es la solución óptima del programa entero.

- El valor de su función objetivo es $Z_{PE} = 8$.

Paso 0 – Inicialización Hacer $\mathcal{L} = \{PE\}$, $S^{(0)} = S$, $\overline{z}^{(0)} = \infty$ y $\underline{z}_{PE} = -\infty$.

Paso 1 – Comprobación de final Si $\mathcal{L} = \emptyset$, la solución $\boldsymbol{x}$ que daba la función objetivo $\underline{z}_{PE} = \boldsymbol{c}^T\boldsymbol{x}$ es la óptima.

Paso 2 – Selección de problema, relajación lineal y resolución Seleccionar y borrar de $\mathcal{L}$ un problema $PE^{(i)}$. Resolver su relajación lineal $PL^{(i)}$. Sea $z_{PL}^{(i)}$ el valor óptimo de la función objetivo de este problema y $\boldsymbol{x}_{PL}^{(i)}$ su solución óptima (si existe).

Paso 3 – Poda Si $z_{PL}^{(i)} \leq \underline{z}_{PE}$ ir al paso 1 (nótese que si la relajación se resuelve por un algoritmo dual, este paso es aplicable tan pronto como el valor de la función objetivo del programa dual se hace inferior a $\underline{z}_{PE}$).
Si $\boldsymbol{x}_{PL}^{(i)} \notin S^{(i)}$ ir al paso 4.
Si $\boldsymbol{x}_{PL}^{(i)} \in S^{(i)}$ y $\boldsymbol{c}^T\boldsymbol{x}_{PL}^{(i)} > \underline{z}_{PE}$, hacer $\underline{z}_{PE} = \boldsymbol{c}^T\boldsymbol{x}_{PL}^{(i)}$. Borrar de $\mathcal{L}$ todos los problemas tales que $\overline{z}^{(i)} \leq \underline{z}_{PE}$. Si $\boldsymbol{c}^T\boldsymbol{x}_{PL}^{(i)} = z_{PL}^{(i)}$ ir al paso 1; si no, al paso 4.

Paso 4 – División Sea $\{S^{(ij)}\}_{j=1}^{k}$ una división de S. Añadir los problemas $\{PE^{(ij)}\}_{j=1}^{k}$ a $\mathcal{L}$, donde $\overline{z}^{(ij)} = z_{PL}^{(i)}$ para $j = 1, \ldots, k$.
Ir al paso 1.

Algoritmo Branch and Bound

- $\mathcal{L}$ designa una colección de programas enteros, $PE^{(i)}$, cada uno tiene la forma $z_{PE}^{(i)} = \text{max.}\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in S^{(i)}\}$, donde $S^{(i)} \subseteq S$.

Branch-and-bound con relajación lineal

- Consideraremos los programas enteros generales

  (PE) max. $\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in S\}$, con $S = \{\boldsymbol{x} \in \mathbb{Z}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}, \boldsymbol{x} \geq \boldsymbol{0}\}$,

  y la forma de resolverlos

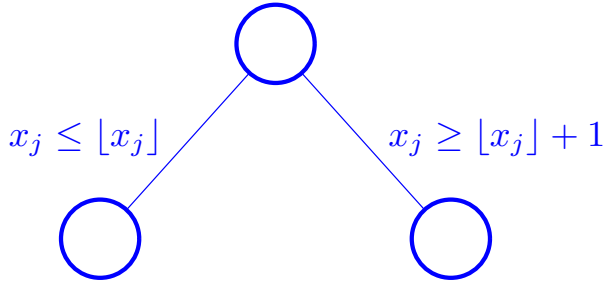
- Mediante esta relajación, el conjunto factible, S, se reemplaza por

$$S_{PL}^{(0)} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x} \geq \boldsymbol{0}, \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}\},$$

  haciéndose en cada relajación $z_{PL}(\boldsymbol{x}) = \boldsymbol{c}^T\boldsymbol{x}$.

División de ramas

- Como se usa relajación lineal, la división se debe hacer con condiciones lineales. La forma más conveniente de hacerlo es como indica la figura.



  Es decir, desde el nudo que define la solución correspondiente, si esta no tiene alguno de sus elementos, j, que ha de ser entero, entero, se hace partir dos ramas: una en la que ese x_j se acote superiormente por su entero inmediato inferior; otra que se acote inferiormente por su entero superior.

- Los dos nuevos programas se pueden resolver mediante el método dual del simplex; el tamaño de la base seguirá igual.

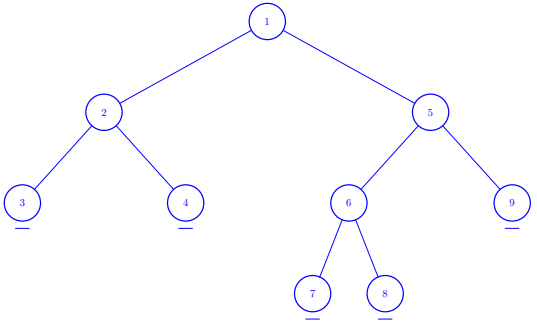
Selección del nudo a estudiar

Proposición Si $P = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b},\ \boldsymbol{x} \geq \boldsymbol{0}\}$ es acotado, el árbol de búsqueda desarrollado a partir de dicotomías de las variables es finito. En particular, si $\omega_j = \lceil \max\{x_j : \boldsymbol{x} \in P\} \rceil$, ningún camino del árbol de búsqueda puede contener más de $\sum_{j \in N} \omega_j$ ramas.

Proposición Si el nudo t del árbol de búsqueda con conjunto de condiciones $S^{(t)}$ es tal que max. $\{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in S_{PL}^{(t)}\} > z_{PE}$, ese nudo no se puede podar o rechazar.

- Dada una lista $\mathcal{L}$ de subárboles del árbol de búsqueda con nudos no rechazados, qué nudo elegir como el próximo a examinar.
- Dos opciones:
  - Analizar uno siguiendo unas reglas preestablecidas; o
  - Escogerlo según la información disponible en ese punto sobre cotas de las variables, número de nudos activos, etc.
- Existen varias reglas preestablecidas que se pueden utilizar. La más extendida es la denominada LIFO (del inglés Last In First Out).

- LIFO: Primero analizar todos los nudos "hijos"de uno determinado y luego volver hacia atrás y escoger otro del mismo nivel del nudo "padre" que no haya sido analizado.



Los nudos subrayados son los podados o rechazados.

- Ventajas:
  - La relajación lineal de un nudo hijo se obtiene de la lineal del nudo padre sin más que añadir una cota en una variable. El método dual del simplex, sin reinvertir la base ni modificar las estructuras de datos sustancialmente, obtiene muy rápidamente la solución que define el nudo hijo.
  - La experiencia dice que las soluciones factibles se encuentran más bien en puntos profundos del árbol de búsqueda.

- Otra regla: Búsqueda en amplitud o anchura. Analizar todos los nudos de un determinado nivel antes de pasar a otro más profundo. En códigos especializados en variables binarias.

- Un código "sencillo" recursivo de **Matlab** para resolver un Programa Entero:

```
function [xstar,Jmin] = ilp_bb_1(c,A,b,xlb,xub,xstar,Jmin)
% Integer linear programming by (recursive) branch & bound
% This algorithm uses linprog.m from the optimisation toolbox
 tol = 1e-5;                           % anything less is considered zero
 if ~exist('xlb','var'), xlb = 0*c; end % canonical form (for the moment)
 if ~exist('xub','var'), xub = Inf*ones(size(c)); end   % Upper bound defaults
 if ~exist('xstar','var'), xstar = []; end
 if ~exist('Jmin','var'),  Jmin = Inf; end              % Try to minimise this
   % Use the following two line if you prefer OPTI Optimization toolbox
   % LPrelax = opti('f',c,'ineq',A,b, 'bounds', xlb, xub);
   % [x,j,exitflag] = solve(LPrelax);
 optns = optimset('display','off');         % turn off diagnostics in Linprog
 [x,j,exitflag] = linprog(c,A,b,[],[],xlb,xub,[],optns);
 if exitflag~=1, return, end                % if infeasible, branch ended
 if j>Jmin, return, end                     % if current cost J=cT x is worse, drop
 idx = find(abs(x-round(x)) > tol);         % which are non-integers ?
 if isempty(idx)                            % All integer solutions
    if j<Jmin, xstar=round(x); Jmin=j; end  % New integer optimum
    return                                  % problem solved
 end
   % Depth first recursion
 xnlb = xlb; xnlb(idx(1)) = ceil(x(idx(1)));        % Take x<--int. greater than x
 [xstar,Jmin] = ilp_bb_1(c,A,b,xnlb,xub,xstar,Jmin);
 xnub = xub; xnub(idx(1)) = floor(x(idx(1)));       % Take x<--int. less than x
 [xstar,Jmin] = ilp_bb_1(c,A,b,xlb,xnub,xstar,Jmin);
end
```

- **Ejemplo** Resolvamos este problema de programación entera

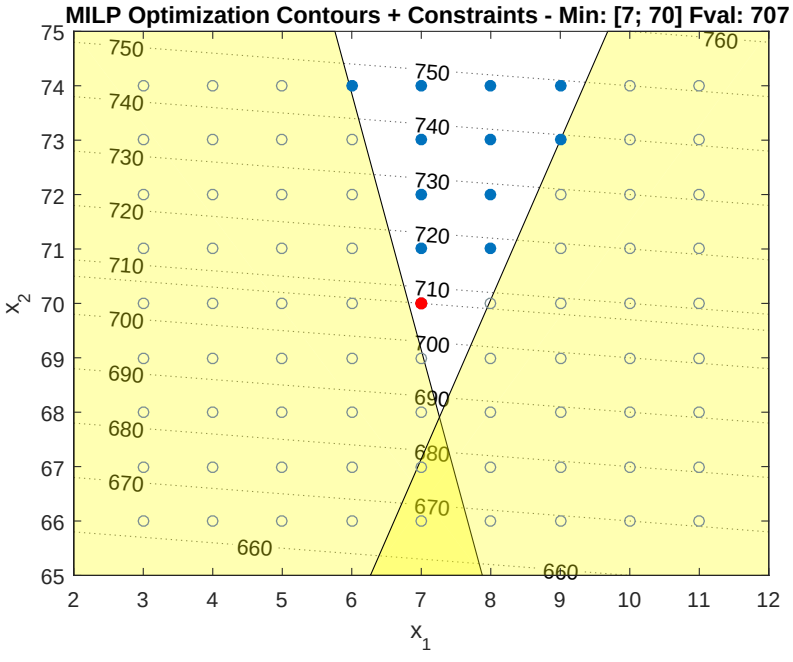
$$\begin{aligned} \text{minimizar} \quad & x_1 + 10x_2 \\ \text{s. a} \quad & 66x_1 + 14x_2 \geq 1430 \\ & -82x_1 + 28x_2 \geq 1306 \\ & x_1,\ x_2 \text{ enteras}, \geq 0. \end{aligned}$$

Con el programa presentado

```
>> c3=[1;10]
>> A3=[-66 -14;82 -28]
>> b3=[-1430;-1306]
>> [xreal,jreal]=linprog(c3,A3,b3)
Optimization terminated.
xreal =
   7.261682243025319
  67.909212283785493
jreal =
    6.863538050808803e+002
>> [xint,j]=ilp_bb_1(c3,A3,b3)
xint =
     7
    70
j =
   707
```

- Con la herramienta OPTI Toolbox

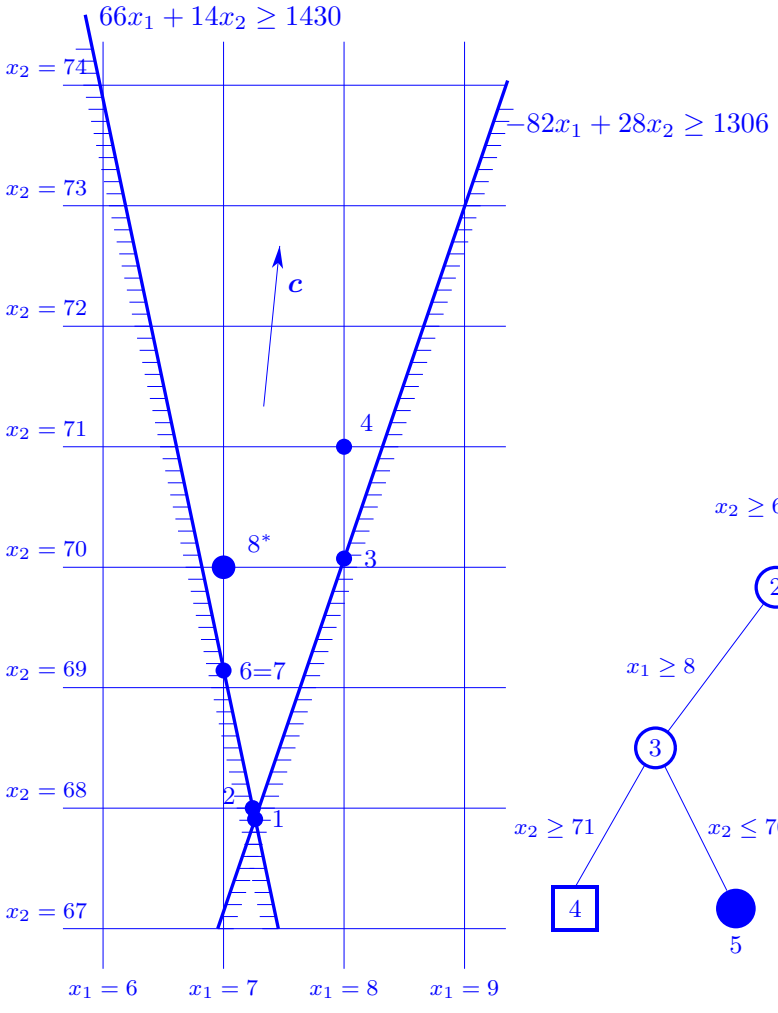
```
>> f1=[1 10]';
>> A1=[-66 -14; 82 -28];
>> b1=[-1430 -1306]';
>> lb = [0;0]; ub = [];
>> xtype = 'II';
>> Opt=opti('f',f1,'ineq',A1,b1,'lb',lb,'xtype',xtype)
-------------------------------------------------------
Mixed Integer Linear Program (MILP) Optimization
 min  f'x
 s.t. rl <= Ax <= ru
      lb <= x <= ub
      xi = Integer / Binary
-------------------------------------------------------
   Problem Properties:
# Decision Variables:        2
# Constraints:               6
  # Linear Inequality:       2
  # Bounds:                  2
  # Integer Variables:       2
-------------------------------------------------------
  Solver Parameters:
Solver:                   CBC
-------------------------------------------------------
>> [xi,fval]=solve(Opt)
xi =
     7
    70
fval =
   707
>> plot(Opt)
```

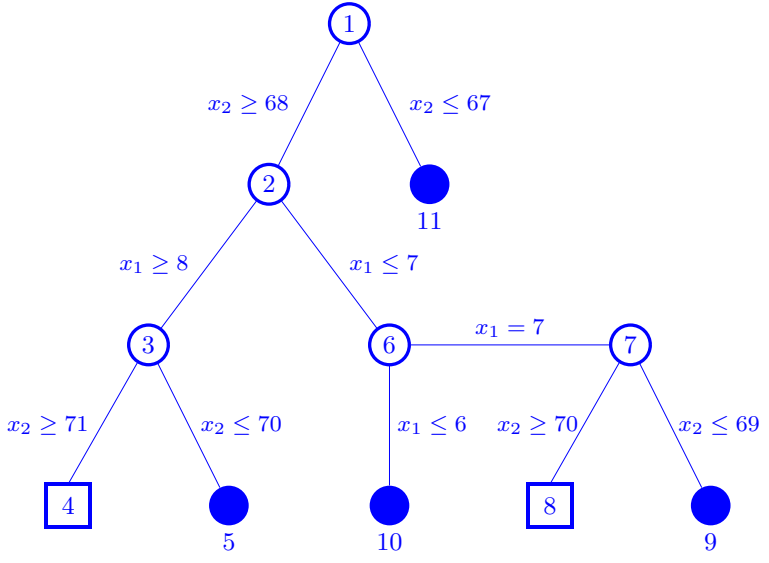


- Resolvámoslo ahora con el software BBMI de programación entera y lineal. El fichero de datos que requeriría este problema es el que sigue

```
PROCESO ITERATIVO
NAME          Ej12.2.3
ROWS
 N  OBJ
 G  FILA1
 G  FILA2
COLUMNS
INT
    X1        OBJ       1              FILA1     66
    X1        FILA2     -82
    X2        OBJ       10             FILA1     14
    X2        FILA2     28
RHS
    RHS       FILA1     1430
    RHS       FILA2     1306
ENDATA
```

- La gráfica de cómo opera BBMI es esta:





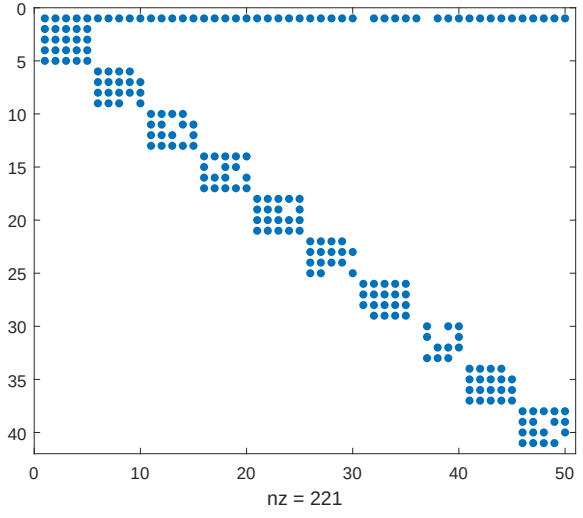
- El resultado impreso de BBMI se lista a continuación.

```
Problema Ej12.2.3


*** Estadísticas del Problema
        3 Fila(s)
        2 Restricción(es)
        2 Variable(s) de decisión
        2 Variable(s) de holgura/artificiales
        6 Elementos no cero
        2 Variable(s) entera(s)
    Densidad de la matriz de coeficientes A: 100.000%


*** Estadísticas de INVERT
        3 elementos no cero en la base
        0 columnas estructurales en la base
        0 vectores columna antes del "bump"
        3 vectores columna después del "bump"
    L:    0 elementos no cero;    0 vectores ETA
    U:    2 elementos no cero;    2 vectores ETA
    Total:    0 elementos no en la diagonal;    2 vectores ETA
    Máximo de transformaciones ETA:    2; máximo número de elementos ETA:    9
    Error relativo en x:   .000000D+00


Ite. Inf. Nopt. Sum.Inf/F.Obj Entra de-a Cost.Red Sale de-a
Paso    Pivote  El.Eta
   1    2    1   .2736000D+04    2  L->B -.42D+02    3 B->L  .10D+03 -.14D+02     2
   2    0    1   .6863538D+03    1  L->B -.46D+02    4 B->L  .73D+01  .21D+03     5

                   --- SOLUCIÓN INICIAL PROGRAMA LINEAL ---
                       --------------------------------

    Nombre del problema: Ej12.2.3
    No. de iteraciones:         3
    Valor de la función objetivo:           686.353805073431


*** FILAS

 No. ..Fila.. en ....Valor.... ...Holgura... .Lí.Inferior. .Lí.Superior. Val.Dual.

   1 OBJ      BS 686.35381    -686.35381       Ninguno       Ninguno     1.000
   2 FILA1    LI 1430.0000     .00000000     1430.0000       Ninguno     .2830
   3 FILA2    LI 1306.0000     .00000000     1306.0000       Ninguno     .2156


*** COLUMNAS

 No. .Columna en ....Valor.... Coste en F.O. .Lí.Inferior. .Lí.Superior. Cos.Red.

   1 X1       BS 7.2616822     1.0000000      .00000000       Ninguno     .000
   2 X2       BS 67.909212     10.000000      .00000000       Ninguno     .000
```

```
 Variable                          Nudos    Variables                  Valor
Separación     Nivel   Dirección   en Lista  Ent. No Ent.
Iteración    Func. Obj.
     2           1       X>  68       1          2          4 D     6.8724242D+02
     1           2       X>   8       2          1          6 D     7.0871429D+02
     2           3       X>  71       3          1          7 D     7.1800000D+02

* Nueva solución entera; z(PE)=        718.00000; Tiempo desde última:    .0001 seg.

                                --- SOLUCIÓN ENTERA ---
                                    ---------------

    Tiempo desde última:    .0001 seg.
    Nombre del problema: Ej12.2.3
    No. de iteraciones:          7
    Valor de la función objetivo:           718.000000000000


*** FILAS

 No. ..Fila.. en ....Valor.... ...Holgura... .Lí.Inferior. .Lí.Superior. Val.Dual.

   1 OBJ      BS 718.00000    -718.00000       Ninguno       Ninguno     1.000
   2 FILA1    BS 1522.0000     92.000000     1430.0000       Ninguno     .1269E-15
   3 FILA2    BS 1332.0000     26.000000     1306.0000       Ninguno     .1349E-16


*** COLUMNAS

 No. .Columna en ....Valor.... Coste en F.O. .Lí.Inferior. .Lí.Superior. Cos.Red.

   1 X1       LIB 8.0000000     1.0000000     8.0000000       Ninguno     1.00
   2 X2       LIB 71.000000     10.000000     71.000000       Ninguno     10.0


 Variable                          Nudos    Variables                  Valor
Separación     Nivel   Dirección   en Lista  Ent. No Ent.
Iteración    Func. Obj.
     2           3       X<  70       2   -Nudo desechado en BKTRAK- 1.0000000D+20
     1           2       X<   7       1          0          9 P     6.9842857D+02
     2FIJ   1    3       X>  70       3          1         10 D     7.0700000D+02

* Nueva solución entera; z(PE)=        707.00000; Tiempo desde última:    .0001 seg.

                                --- SOLUCIÓN ENTERA ---
                                    ---------------

    Tiempo desde última:    .0001 seg.
    Nombre del problema: Ej12.2.3
    No. de iteraciones:         10
    Valor de la función objetivo:           707.000000000000


*** FILAS
```

```
 No. ..Fila.. en ....Valor.... ...Holgura... .Lí.Inferior. .Lí.Superior. Val.Dual.

   1 OBJ      BS 707.00000    -707.00000       Ninguno       Ninguno     1.000
   2 FILA1    BS 1442.0000     12.000000     1430.0000       Ninguno     .1269E-15
   3 FILA2    BS 1386.0000     80.000000     1306.0000       Ninguno     .1349E-16


*** COLUMNAS

 No. .Columna en ....Valor.... Coste en F.O. .Lí.Inferior. .Lí.Superior. Cos.Red.

   1 X1      EQB 7.0000000     1.0000000     7.0000000       Ninguno     1.00
   2 X2      LIB 70.000000     10.000000     70.000000       Ninguno     10.0


 Variable                           Nudos    Variables                   Valor
Separación     Nivel   Dirección   en Lista  Ent. No Ent.
Iteración    Func. Obj.
     2            3       X<  69        2   -Nudo desechado en BKTRAK- 1.0000000D+20
     1            3       X<   6        1   -Nudo desechado en BKTRAK- 7.4457143D+02
     2            1       X<  67        0   -Nudo desechado en BKTRAK- 1.0000000D+20

                        --- SOLUCIÓN ENTERA OPTIMA ---
                            ----------------------

    Nombre del problema: Ej12.2.3
    No. de iteraciones:        10
    Valor de la función objetivo:           707.000000000000


*** FILAS

 No. ..Fila.. en ....Valor.... ...Holgura... .Lí.Inferior. .Lí.Superior. Val.Dual.

   1 OBJ      BS 707.00000    -707.00000       Ninguno       Ninguno     1.000
   2 FILA1    BS 1442.0000     12.000000     1430.0000       Ninguno     .1269E-15
   3 FILA2    BS 1386.0000     80.000000     1306.0000       Ninguno     .1349E-16


*** COLUMNAS

 No. .Columna en ....Valor.... Coste en F.O. .Lí.Inferior. .Lí.Superior. Cos.Red.

   1 X1      EQB 7.0000000     1.0000000     7.0000000       Ninguno     .000
   2 X2      LIB 70.000000     10.000000     70.000000       Ninguno     10.0

Tiempo total de CPU en cálculos:    .0603 segundos
```

- Para darnos una idea de las prestaciones de estos algoritmos, resolvamos un problema relativamente difícil de Programación Entera con 50 variables, todas enteras, y cuya matriz de condiciones tiene este esquema:



- Lo vamos a resolver con el solver de PE de `OPTI Toolbox` y con `ilp_bb_1`

- Con Intlinprog de Matlab:

```
>> problem = mpsread('DEMOS5_noint.mps')
problem =
          f: [50x1 double]
      Aineq: [41x50 double]
      bineq: [41x1 double]
        Aeq: [0x50 double]
        beq: [0x1 double]
         lb: [50x1 double]
         ub: [50x1 double]
     intcon: [50x1 double]
     solver: 'intlinprog'
    options: [1x1 optim.options.Intlinprog]
>> options = optimoptions('intlinprog','Display','final','PlotFcns',@optimplotmilp);
>> problem.options = options;
>> tic,[x,fval,exitflag,output] = intlinprog(problem),toc
LP:                Optimal objective value is -1171.225417.

Cut Generation:    Applied 5 Gomory cuts, 10 clique cuts, and 5 cover cuts.
                   Lower bound is -1046.000000.
                   Relative gap is 0.00%.
Optimal solution found.

Intlinprog stopped at the root node ..

x =
    1.0000
    1.0000
    1.0000
    1.0000
         0
    1.0000
    1.0000
         0
    1.0000
         0
         0
    1.0000
    1.0000
    1.0000
    1.0000
         0
    1.0000
    1.0000
    1.0000
         0
         0
```

```
    1.0000
    1.0000
         0
    1.0000
    1.0000
         0
    1.0000
    1.0000
    1.0000
    1.0000
    1.0000
         0
    1.0000
         0
         0
         0
         0
         0
    1.0000
         0
         0
    1.0000
         0
         0
         0
    1.0000
         0
    1.0000
    1.0000

fval =

  -1.0460e+03


exitflag =

     1


output =

        relativegap: 0
        absolutegap: 0
      numfeaspoints: 1
           numnodes: 0
    constrviolation: 1.5543e-15
            message: 'Optimal solution found'

Elapsed time is 0.446243 seconds.
>>
```

- Con OPTI Toolbox:

```
>> prob_1 = coinRead('DEMOS5_noint.mps');
>> spy(prob_1.A);
>> Opt_1 = opti('f',-prob_1.f,'ineq',prob_1.A,prob_1.ru,'bounds',prob_1.lb,prob_1.ub,'xtype',xtype)
-------------------------------------------------------
Mixed Integer Linear Program (MILP) Optimization
 min  f'x
 s.t. rl <= Ax <= ru
      lb <= x <= ub
      xi = Integer / Binary
-------------------------------------------------------
   Problem Properties:
# Decision Variables:       50
# Constraints:              191
  # Linear Inequality:       41
  # Bounds:                  100
  # Integer Variables:       50
-------------------------------------------------------
  Solver Parameters:
Solver:                     CBC
-------------------------------------------------------
>> tic,[x,fval,exitflag,info] = solve(Opt_1),toc
x =
    1.0000
    1.0000
    1.0000
    1.0000
         0
    1.0000
    1.0000
         0
    1.0000
         0
         0
    1.0000
    1.0000
    1.0000
    1.0000
         0
    1.0000
    1.0000
    1.0000
         0
         0
    1.0000
    1.0000
         0
    1.0000
    1.0000
         0
```

```
    1.0000
    1.0000
    1.0000
    1.0000
    1.0000
         0
    1.0000
         0
         0
         0
         0
         0
    1.0000
         0
         0
    1.0000
         0
         0
         0
    1.0000
         0
    1.0000
    1.0000

fval =
       -1046
exitflag =

     1
info =
        Nodes: 2
       AbsGap: 24.9233
       RelGap: 0.0238
         Time: 0.0581
    Algorithm: 'CBC: Branch and Cut using CLP'
       Status: 'Integer Optimal'

 Elapsed time is 0.060537 seconds.
>>
```

- Con `ilp_bb_1`:

```
>> tic,[xint,j]=ilp_bb_1(-prob.f,A,prob_1.ru,prob_1.lb,prob_1.ub),toc
xint =
     1
     1
     1
     1
     0
     1
     1
     0
     1
     0
     0
     1
     1
     1
     1
     0
     1
     1
     1
     0
     0
     1
     1
     0
     1
     1
     0
     1
     1
     1
     1
     1
     0
     1
     0
     0
     0
     0
     0
     1
     0
     0
     1
     0
     0
     0
     1
     0
     1
     1
j =

  -1.0460e+03
Elapsed time is 7295.551641 seconds.
>>
```

El tiempo empleado es inaceptable. La estrategia es mala al no considerar ninguna forma de ahorrarse visitar todos los nudos del árbol de búsqueda.

Selección de la variable de ramificación

- Es evidente que para mejorar las prestaciones de un algoritmo Branch and Bound es necesario hacer algo que evite visitar todos los nudos del árbol de búsqueda de soluciones enteras. Veamos lo que hace BBMI.

- Supongamos que se ha elegido i como el nudo que se va a analizar. Asociado a ese nudo habrá un programa lineal de solución $\boldsymbol{x}^{(i)}$,

$$x_{B_i} = \bar{a}_{i0} + \sum_{j=m+1}^{n} \bar{a}_{ij}(-x_j) \qquad \text{para } i = 1, \ldots, m;$$

  la función objetivo es $x_0 = \bar{a}_{00} + \sum_{j=m+1}^{n} \bar{a}_{0j}(-x_j)$. Los $\bar{a}_{0j}$, $j = m+1, \ldots, n$, son los costes reducidos de las variables no básicas.

- El paso siguiente de escoger la variable que va a definir la ramificación o división que parta de i es crítico para las prestaciones del procedimiento. Apuntemos alguna estrategia de mejora.

Penalizaciones de variables básicas

- Supongamos que en el nudo i alguna variable básica x_p no es entera debiendo serlo, es decir

$$x_p = \bar{a}_{p0} + \sum_{j=m+1}^{n} \bar{a}_{pj}(-x_j) = n_{p0} + f_{p0},$$

donde n_{p0} designa la parte entera de x_p, $\lfloor x_p \rfloor$, y f_{p0} la parte fraccionaria.[3]

- La división que definirá la variable x_p es

$$x_p \geq n_{p0} + 1 \quad \text{y} \quad x_p \leq n_{p0}.$$

[3] Recordemos $\lfloor 3, 22 \rfloor = 3$; $\lfloor -4, 12 \rfloor = -5$.

- Del algoritmo dual del simplex podemos deducir que la imposición de la nueva cota $x_p \geq n_{p0} + 1$ a la variable x_p implicará un empeoramiento (reducción) del valor de la función objetivo.
  - Como la variable básica x_p, realizando una iteración del método dual del simplex, pasaría como mínimo a ser no básica, otra variable no básica, x_j, que estuviese en uno de sus límites, se incrementaría o decrementaría según estuviese en el inferior o superior.
  - Es decir, se reduciría el valor de la función objetivo en una cantidad o penalización que vendría dada por la expresión

$$P_U = \begin{cases} \text{mín}_{j,\bar{a}_{pj}<0}(1-f_{p0})\frac{\bar{a}_{0j}}{-\bar{a}_{pj}}, & \text{si } x_j = l_j, \text{ o por} \\ \text{mín}_{j,\bar{a}_{pj}>0}(1-f_{p0})\frac{\bar{a}_{0j}}{-\bar{a}_{pj}}, & \text{si } x_j = u_j. \end{cases}$$

- Razonando de manera similar, la imposición de la nueva cota $x_p \leq n_{p0}$ a la variable x_p implicará un empeoramiento (reducción) del valor de la función objetivo que vendrá dado por la expresión

$$P_D = \begin{cases} \min_{j,\bar{a}_{pj}>0} f_{p0}\frac{\bar{a}_{0j}}{\bar{a}_{pj}}, & \text{si } x_j = l_j, \\ \min_{j,\bar{a}_{pj}<0} f_{p0}\frac{\bar{a}_{0j}}{\bar{a}_{pj}}, & \text{si } x_j = u_j. \end{cases}$$

- Cualquier solución entera que se pudiese obtener partiendo del nudo i estaría por consiguiente acotada superiormente por

$$\max\{x_0 - P_U,\ x_0 - P_D\}.$$

- Se podrá conseguir una mejor solución desde el nudo i si y sólo si

$$\min\{P_U,\ P_D\} < x_0 - \underline{z}_{PE}$$

para cada variable básica que no sea entera debiendo serlo.
Si esta condición no se cumple se abandona el nudo i.

- Si todas las variables x_p cumplen esa condición, la cuestión siguiente a solventar es qué variable de ramificación se elige.
  - Se puede elegir aquella con la penalización asociada más pequeña.
  - O escoger aquella variable con una penalización asociada más grande y comenzar imponiendo el límite opuesto al que determina esa penalización.

- Si, por ejemplo, la penalización más grande asociada a una variable x_p es P_U, comenzar analizando el problema que resulte de imponer a esa variable la nueva cota n_{p0} e incluir en la lista de nudos a analizar más adelante el que resulte de imponer la cota $n_{p0} + 1$.

- Si se almacenan las peores soluciones, una vez que se encuentre una buena, se rechazarán rápidamente buena parte de los nudos que queden en la lista.

Penalizaciones de variables no básicas

- Posibles cambios en variables no básicas que impondrían nuevas cotas en el nudo i. Cualquier solución entera estaría acotada por

$$\text{máx}\{x_0 - P_U^*,\ x_0 - P_D^*\},$$

donde ahora

$$P_U^* = \begin{cases} \text{mín}_{j,\bar{a}_{pj}<0} \left\{ \begin{array}{ll} \bar{a}_{0j}(1-f_{p0})/(-\bar{a}_{pj}), & j \in M \\ \text{máx}\{\bar{a}_{0j},\ \bar{a}_{0j}(1-f_{p0})/(-\bar{a}_{pj})\}, & j \in J \end{array} \right\} & \text{si } x_j = l_j \\ \text{mín}_{j,\bar{a}_{pj}>0} \left\{ \begin{array}{ll} \bar{a}_{0j}(1-f_{p0})/(-\bar{a}_{pj}), & j \in M \\ \text{máx}\{\bar{a}_{0j},\ \bar{a}_{0j}(1-f_{p0})/(-\bar{a}_{pj})\}, & j \in J \end{array} \right\} & \text{si } x_j = u_j \end{cases}$$

y

$$P_D^* = \begin{cases} \text{mín}_{j,\bar{a}_{pj}>0} \left\{ \begin{array}{ll} \bar{a}_{0j} f_{p0}/\bar{a}_{pj}, & j \in M \\ \text{máx}\{\bar{a}_{0j},\ \bar{a}_{0j} f_{p0}/\bar{a}_{pj}\}, & j \in J \end{array} \right\} & \text{si } x_j = l_j \\ \text{mín}_{j,\bar{a}_{pj}<0} \left\{ \begin{array}{ll} \bar{a}_{0j} f_{p0}/\bar{a}_{pj}, & j \in M \\ \text{máx}\{\bar{a}_{0j},\ \bar{a}_{0j} f_{p0}/\bar{a}_{pj}\}, & j \in J \end{array} \right\} & \text{si } x_j = u_j. \end{cases}$$

J es el conjunto de índices de las variables no básicas que han de ser enteras y M el de las no básicas que no han de ser enteras.

- El nudo i se podrá desechar si, para cualquier x_p a la que se introducen nuevas cotas,

$$\text{mín}\{P_U^*,\ P_D^*\} \geq x_0 - \underline{z}_{PE}.$$

- Este criterio no es el único; recordemos el corte de Gomory asociado a x_p,

$$-f_{p0} + \sum_{j=m+1}^{n} f_{pj} x_j \geq 0,$$

que se habría de satisfacer en cualquier solución que se obtuviese partiendo del problema actual.

- Asociado a ese corte de Gomory se puede derivar la penalización

$$P_G^* = \begin{cases} \min_j \left\{ \begin{array}{lll} f_{p0}\bar{a}_{0j}/\bar{a}_{pj} & \text{si } \bar{a}_{pj} \geq 0 & j \in M \\ (1-f_{p0})\bar{a}_{0j}/(-\bar{a}_{pj}) & \text{si } \bar{a}_{pj} < 0 & j \in M \\ f_{p0}\bar{a}_{0j}/f_{pj} & \text{si } f_{pj} \leq f_{p0} & j \in J \\ (1-f_{p0})\bar{a}_{0j}/(1-f_{pj}) & \text{si } f_{pj} > f_{p0} & j \in J \end{array} \right\} & \text{si } x_j = l_j; \\ \min_j \left\{ \begin{array}{lll} f_{p0}\bar{a}_{0j}/\bar{a}_{pj} & \text{si } \bar{a}_{pj} \leq 0 & j \in M \\ (1-f_{p0})\bar{a}_{0j}/(-\bar{a}_{pj}) & \text{si } \bar{a}_{pj} > 0 & j \in M \\ f_{p0}(-\bar{a}_{0j})/f_{pj} & \text{si } f_{pj} \leq f_{p0} & j \in J \\ (1-f_{p0})(-\bar{a}_{0j})/(1-f_{pj}) & \text{si } f_{pj} > f_{p0} & j \in J \end{array} \right\} & \text{si } x_j = u_j. \end{cases}$$

- Se puede comprobar que

$$P_G^* \geq \min(P_U^*, P_D^*),$$

lo que trae como consecuencia que el nudo i se podrá desechar si, para cualquier x_p a la que se introducen nuevas cotas,

$$P_G^* \geq x_0 - \underline{z}_{PE}.$$

- **Ejemplo** Volvamos al ejemplo: maximizar $7x_1 + 2x_2$

$$\begin{aligned} \text{s. a} \quad -x_1 + 2x_2 &\leq 4 \\ 5x_1 + x_2 &\leq 20 \\ -2x_1 - 2x_2 &\leq -7 \\ \boldsymbol{x} &\in \mathbb{Z}_+^2. \end{aligned}$$

- Con las variables de holgura x_3, x_4 y x_5, la relajación lineal da la solución

$$\begin{aligned} x_0 + \tfrac{3}{11}x_3 + \tfrac{16}{11}x_4 &= \tfrac{332}{11} \\ x_1 - \tfrac{1}{11}x_3 + \tfrac{2}{11}x_4 &= \tfrac{36}{11} \\ x_2 + \tfrac{5}{11}x_3 + \tfrac{1}{11}x_4 &= \tfrac{40}{11} \\ \tfrac{8}{11}x_3 + \tfrac{6}{11}x_4 + x_5 &= \tfrac{75}{11}, \end{aligned}$$

  donde $x_3 = x_4 = 0$.
  De aquí que $z_{PL}^{(0)} = x_0 = 332/11$ y $\boldsymbol{x}^{(0)} = [36/11, 40/11, 0, 0, 75/11]^T$.

- Trabajemos con las penalizaciones que incluye BBMI. Las variables básicas son x_1, x_2 y x_5. Esta última no se requiere que sea entera. Las variables no básicas de la solución obtenida están todas en su límite inferior.

- Para empezar, como todavía no tenemos ninguna solución factible de z_{PE}, determinemos las penalizaciones por bifurcar a:

$$x_1 \leq 3: \quad P_{1D}^{0-} = \text{mín}_{j,\bar{a}_{pj}>0}\{\bar{a}_{0j} f_{p0}/\bar{a}_{pj},\ j \in M\} = \quad \text{mín}\left\{\frac{16}{11}\cdot\frac{3}{11}\Big/\frac{2}{11}\right\} = \frac{24}{11};$$

$$x_1 \geq 4: \quad P_{1U}^{0+} = \text{mín}_{j,\bar{a}_{pj}<0}\{\bar{a}_{0j}(1-f_{p0})/(-\bar{a}_{pj}),\ j \in M\} = \text{mín}\left\{\frac{3}{11}\cdot(1-\frac{3}{11})\Big/\frac{1}{11}\right\} = \frac{24}{11};$$

$$x_2 \leq 3: \quad P_{2D}^{0-} = \text{mín}_{j,\bar{a}_{pj}>0}\{\bar{a}_{0j} f_{p0}/\bar{a}_{pj},\ j \in M\} = \quad \text{mín}\left\{\frac{16}{11}\cdot\frac{7}{11}\Big/\frac{1}{11}, \frac{3}{11}\cdot\frac{7}{11}\Big/\frac{5}{11}\right\} = \frac{21}{55}.$$

- La penalización P_{2D}^{0+} la podemos considerar ∞ pues de la solución del programa lineal anterior se observa que al incrementar la variable x_2, x_3 y x_4 decrecen.

- De las penalizaciones calculadas, la mejor (menos mala) se obtiene al hacer $x_2 \leq 3$. Añadamos esa condición, es decir, $x_2 = \frac{40}{11} - \frac{5}{11}x_3 - \frac{1}{11}x_4 \leq 3$.

- La relajación lineal resultante anterior más esta condición queda:

$$\begin{array}{rrrrrrrrr}
x_0 & & + \frac{3}{11}x_3 & + \frac{16}{11}x_4 & & & = & \frac{332}{11} \\
& x_1 & - \frac{1}{11}x_3 & + \frac{2}{11}x_4 & & & = & \frac{36}{11} \\
& & x_2 + \frac{5}{11}x_3 & + \frac{1}{11}x_4 & & & = & \frac{40}{11} \\
& & & \frac{8}{11}x_3 & + \frac{6}{11}x_4 & + x_5 & = & \frac{75}{11} \\
& & - \frac{5}{11}x_3 & - \frac{1}{11}x_4 & & + x_6 & = & -\frac{7}{11}.
\end{array}$$

- Después de una iteración del método dual del simplex se llega a la solución óptima siguiente:

$$\begin{array}{rrrrrrrrr}
x_0 & & + \frac{7}{5}x_4 & & + \frac{3}{5}x_6 & = & \frac{149}{5} \\
& x_1 & + \frac{1}{5}x_4 & & - \frac{1}{5}x_6 & = & \frac{17}{5} \\
& & x_2 & & + x_6 & = & 3 \\
& & & \frac{2}{5}x_4 + x_5 & + \frac{8}{5}x_6 & = & \frac{29}{5} \\
& & x_3 + \frac{1}{5}x_4 & & - \frac{11}{5}x_6 & = & \frac{7}{5}\ .
\end{array}$$

De aquí que $z_{PL}^{(1)} = x_0 = 149/5$ y $\boldsymbol{x}^{(1)} = [17/5, 3, 7/5, 0, 29/5]^T$, con $x_6 = 0$. La única variable que debe ser entera en esta solución y no lo es x_1.

- Calculemos ahora las penalizaciones por bifurcar a:

$$x_1 \leq 3: \quad P_{1D}^{1-} = \min_{j,\bar{a}_{pj}>0}\{\bar{a}_{0j} f_{p0}/\bar{a}_{pj},\ j \in M\} = \min\left\{\frac{7}{5}\cdot\frac{2}{5}\Big/\frac{1}{5}\right\} = \frac{14}{5}$$

$$x_1 \geq 4: \quad P_{1U}^{1+} = \min_{j,\bar{a}_{pj}<0}\{\bar{a}_{0j}(1-f_{p0})/(-\bar{a}_{pj}),\ j \in M\} = \min\left\{\frac{3}{5}\cdot(1-\frac{2}{5})\Big/\frac{1}{5}\right\} = \frac{9}{5}.$$

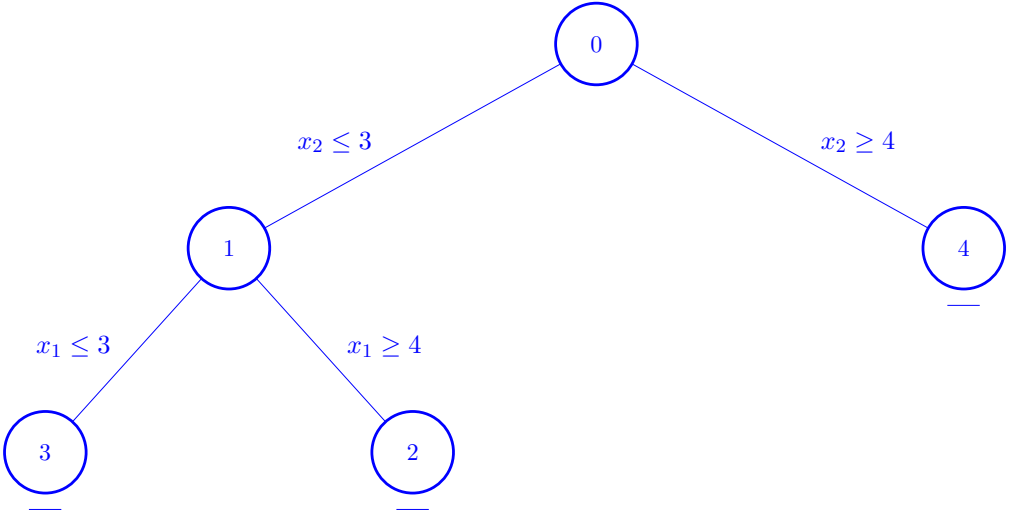
De estas penalizaciones la mejor se obtiene al hacer $x_1 \geq 4$. Añadamos la condición $x_1 = \frac{17}{5} - \frac{1}{5}x_4 + \frac{1}{5}x_6 \geq 4$.

- La relajación lineal resultante anterior más esta condición queda:

$$\begin{array}{rrrrrrrrr}
x_0 & & + \frac{7}{5}x_4 & & + \frac{3}{5}x_6 & & & = & \frac{149}{5} \\
& x_1 & + \frac{1}{5}x_4 & & - \frac{1}{5}x_6 & & & = & \frac{17}{5} \\
& & x_2 & & + x_6 & & & = & 3 \\
& & & \frac{2}{5}x_4 + x_5 & + \frac{8}{5}x_6 & & & = & \frac{29}{5} \\
& & x_3 + \frac{1}{5}x_4 & & - \frac{11}{5}x_6 & & & = & \frac{7}{5} \\
& & - \frac{1}{5}x_4 & & + \frac{1}{5}x_6 & - x_7 & & = & \frac{3}{5}\ .
\end{array}$$

- Otra iteración del dual del simplex obtiene la primera solución factible entera $\boldsymbol{x}^{(2)} = [4,\ 0,\ 8,\ 0,\ 1]^T$, con $z_{PE} = 28$.

- Después se analizaría el nudo 3, rechazándose inmediatamente pues del cálculo anterior de penalizaciones se vería que la función objetivo obtenible sería $149/5 - P_{1D}^{1-} = 135/5 = 27 < 28$. Posteriormente también se rechazaría el nudo 4.



Solución entera.
$x^{(2)} = [4,\ 0,\ 8,\ 0,\ 1]^T$
$z_{PL}^{(2)} = \underline{z}_{PE} = z_{PE} = 28$

Programación No lineal en variables Enteras

- La dificultad de estos problemas aumenta considerablemente, pero nada impide utilizar la estrategia Branch and Bound utilizando un *solver* no lineal –fmincon por ejemplo– en vez del lineal que hemos presentado. El código de antes adaptado para resolver un Programa Entero no lineal es éste:

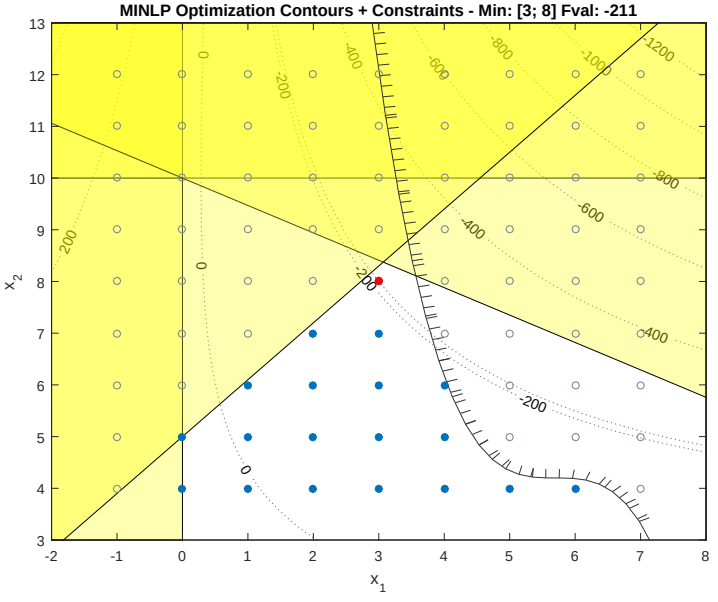
```
function [xstar,Jmin] = inlp_bb(fun,x0,A,b,xlb,xub,NLCon,xstar,Jmin)
% [xstar,Jmin] = ilp_bb(fun,x0,A,b,xlb,xub,NLCon,xstar,Jmin)
% Integer Non-linear programming by (recursive) branch & bound
% Uses fmincon.m from the optimisation toolbox to solve the relaxed problem
 optns = optimset('display','off'); tol = 1e-3; % anything less is zero
 if ~exist('xlb','var'),   xlb = 0*x0; end
 if ~exist('xub','var'),   xub = Inf*ones(size(x0)); end
 if ~exist('xstar','var'), xstar = []; end
 if ~exist('Jmin','var'),  Jmin = Inf; end
   % Now solve relaxed NL constrained problem
 [x,j,exitflag] = fmincon(fun,x0,A,b,[],[],xlb,xub,NLCon,optns);
 if exitflag <1; return, end       % disp(exitflag) if infeasible or not OK
 if j>Jmin + tol; return, end                  % for robustness; truncate
 idx = find(abs(x-round(x)) > tol);            % which are non-integers?
 if isempty(idx)                               % All integer solutions
    if j<Jmin, xstar=round(x); Jmin=j; end     % New integer optimum
    return                                     % problem solved
 end
   % Depth first recursion
 xnlb = xlb; xnlb(idx(1)) = ceil(x(idx(1)));   % new lower bound
 [xstar,Jmin] = inlp_bb(fun,x,A,b,xnlb,xub,NLCon,xstar,Jmin);
 xnub = xub; xnub(idx(1)) = floor(x(idx(1)));
 [xstar,Jmin] = inlp_bb(fun,x,A,b,xlb,xnub,NLCon,xstar,Jmin);
end
```

- Resolviendo

$$\begin{aligned}\text{minimizar } & (x_1-5)^2+(x_2-7)^2-x_1x_2\,(x_2+1)\\ \text{sujeta a } \quad & 0{,}4\,(x_1-5{,}7)^3-4{,}2+x_2\le 0\\ & 0{,}53x_1+x_2\le 10\\ & -1{,}1x_1+x_2\le 5.\end{aligned}$$

Partiremos del punto $\boldsymbol{x}_0=[2{,}5,\ 1{,}5]^T$.

```
>> A=[0.53 1; -1.1 1]; b=[10,5]';
>> lb=[0;0]; ub=[10;10];
>> x0=[2.5,1.5]';
>> fun1=@(x) (x(1)-5).^2+(x(2)-7).^2-x(1)*x(2)*(x(2)+1);
>> [xstar,Jmin]=inlp_bb(fun1,x0,A,b,lb,ub,@circlecon)
xstar =

     3
     8

Jmin =

  -211

>> plot(Opt)
```



La rutina con la condición no lineal es

```
function [c,ceq] = circlecon(x)
 c = 0.4*(x(1)-5.7)^3+x(2)-4.2;
 ceq = [];
end
```

- Con la herramienta OPTI Toolbox

```
>> fun=@(x) (x(1)-5).^2+(x(2)-7).^2-x(1)*x(2)*(x(2)+1);
>> A=[0.53 1; -1.1 1]; b=[10,5]';
>> lb=[0;0]; ub=[10;10];
>> nlcon=@(x) 0.4*(x(1)-5.7)^3+x(2);
>> nlrhs=4.2;
>> x0=[2.5,1.5]';
>> xtype = 'II';
>> Opt=opti('fun',fun,'ineq',A,b,'bounds',lb,ub,'nlcon',nlcon,'nlrhs',nlrhs,'xtype',xt
>> [x,fval] = solve(Opt,x0)

x =

     3
     8

fval =

  -211
>> plot(Opt)
```